Report Number: WUCSE-2008-9

2008-01-01

A Holistic Approach to Decentralized Structural Damage Localization Using Wireless Sensor Networks

Gregory Hackmann, Fei Sun, Nestor Castaneda, Chenyang Lu, and Shirley Dyke

Wireless sensor networks (WSNs) have become an increasingly compelling platform for Structural Health Monitoring (SHM) applications, since they can be installed relatively inexpensively onto existing infrastructure. Existing approaches to SHM in WSNs typically address computing system issues or structural engineering techniques, but not both in conjunction. In this paper, we propose a holistic approach to SHM that integrates a decentralized computing architecture with the Damage Localization Assurance Criterion algorithm. In contrast to centralized approaches that require transporting large amounts of sensor data to a base station, our system pushes the execution of portions of the damage localization algorithm onto... **Read complete abstract on page 2.**

A Holistic Approach to Decentralized Structural Damage Localization Using Wireless Sensor Networks

Gregory Hackmann, Fei Sun, Nestor Castaneda, Chenyang Lu, and Shirley Dyke

Complete Abstract:

Wireless sensor networks (WSNs) have become an increasingly compelling platform for Structural Health Monitoring (SHM) applications, since they can be installed relatively inexpensively onto existing infrastructure. Existing approaches to SHM in WSNs typically address computing system issues or structural engineering techniques, but not both in conjunction. In this paper, we propose a holistic approach to SHM that integrates a decentralized computing architecture with the Damage Localization Assurance Criterion algorithm. In contrast to centralized approaches that require transporting large amounts of sensor data to a base station, our system pushes the execution of portions of the damage localization algorithm onto the sensor nodes, reducing communication costs by two orders of magnitude in exchange for moderate additional processing on each sensor. We present a prototype implementation of this system built using the TinyOS operating system running on the Intel Imote2 sensor network platform. Experiments conducted using two different physical structures demonstrate our system's ability to accurately localize structural damage. We also demonstrate that our decentralized approach reduces latency by 65.5% and energy consumption by 70.4% compared to a typical centralized solution, achieving a projected lifetime of 193 days using three standard AAA batteries. Our work demonstrates the advantages of a holistic approach to cyber-physical systems that closely integrates the design of computing systems and physical engineering techniques.

2008-9

A Holistic Approach to Decentralized Structural Damage Localization Using Wireless Sensor Networks

Authors: Gregory Hackmann, Fei Sun, Nestor Castaneda, Chenyang Lu, and Shirley Dyke

Abstract: Wireless sensor networks (WSNs) have become an increasingly compelling platform for Structural Health Monitoring (SHM) applications, since they can be installed relatively inexpensively onto existing infrastructure. Existing approaches to SHM in WSNs typically address computing system issues or structural engineering techniques, but not both in conjunction. In this paper, we propose a holistic approach to SHM that integrates a decentralized computing architecture with the Damage Localization Assurance Criterion algorithm. In contrast to centralized approaches that require transporting large amounts of sensor data to a base station, our system pushes the execution of portions of the damage localization algorithm onto the sensor nodes, reducing communication costs by two orders of magnitude in exchange for moderate additional processing on each sensor. We present a prototype implementation of this system built using the TinyOS operating system running on the Intel Imote2 sensor network platform. Experiments conducted using two different physical structures demonstrate our system's ability to accurately localize structural damage. We also demonstrate that our decentralized approach reduces latency by 65.5% and energy consumption by 70.4% compared to a typical centralized solution, achieving a projected lifetime of 193 days using three standard AAA batteries. Our work demonstrates the advantages of a holistic approach to cyber-physical systems that closely integrates the design

Type of Report: Other

A Holistic Approach to Decentralized Structural Damage Localization Using Wireless Sensor Networks

Gregory Hackmann*, Fei Sun*, Nestor Castaneda†, Chenyang Lu*, Shirley Dyke†
*Department of Computer Science and Engineering
†Department of Mechanical, Aerospace and Structural Engineering
Washington University in St. Louis

***Abstract*—Wireless sensor networks (WSNs) have become an increasingly compelling platform for Structural Health Monitoring (SHM) applications, since they can be installed relatively inexpensively onto existing infrastructure. Existing approaches to SHM in WSNs typically address computing system issues *or* structural engineering techniques, but not both in conjunction. In this paper, we propose a *holistic* approach to SHM that integrates a decentralized computing architecture with the Damage Localization Assurance Criterion algorithm. In contrast to centralized approaches that require transporting large amounts of sensor data to a base station, our system pushes the execution of portions of the damage localization algorithm onto the sensor nodes, reducing communication costs by two orders of magnitude in exchange for moderate additional processing on each sensor. We present a prototype implementation of this system built using the TinyOS operating system running on the Intel Imote2 sensor network platform. Experiments conducted using two different physical structures demonstrate our system's ability to accurately localize structural damage. We also demonstrate that our decentralized approach reduces latency by 65.5% and energy consumption by 70.4% compared to a typical centralized solution, achieving a projected lifetime of 193 days using three standard AAA batteries. Our work demonstrates the advantages of a holistic approach to cyber-physical systems that closely integrates the design of computing systems and physical engineering techniques.**

I. Introduction

Structural Health Monitoring (SHM) is a promising technique to determine the condition of a civil structure, provide spatial and quantitative information regarding structural damage, or predict the performance of the structure during its lifecycle. Recent years have seen growing interest in SHM based on wireless sensor networks (WSNs) due to their potential to monitor a structure at unprecedented temporal and spatial granularity. However, there remain significant research challenges in SHM. Specifically, a SHM system must (1) detect and localize damages in *complex* structures; (2) provide both *long-term* monitoring and *rapid* analysis in response to severe events (e.g., earthquakes and hurricanes); and (3) meet the stringent *resource and energy constraints* of WSNs.

SHM applications are characteristic examples of complex cyber-physical systems where neither the "cyber" aspects nor the "physical" aspects can adequately be considered in isolation. Previous work in the WSN field primarily addresses system issues like data acquisition and communication, while previous work in the structural engineering field has primarily focused on developing algorithms for damage detection and localization. The separation of computing system design and SHM techniques may result in suboptimal system solutions. For example, existing systems developed in the WSN field usually assume a centralized approach that transports large amounts of data from sensors to a base station. Despite considerable research on network protocols optimized for SHM applications, centralized architectures inherently entail significant communication and energy overhead for data collection. For example, a state-of-art system deployed at the Golden Gate Bridge required 9 hours to collect a single round of data from 64 sensors, resulting in a system lifetime of 10 weeks when using four 6V lantern batteries as a power source [1]. On the other hand, while the structural engineering field has proposed damage detection and localization algorithms that are potentially suitable for decentralized processing, prior research in the field usually does not focus on the design of computing system architectures for implementing such algorithms on WSNs.

We therefore propose a *holistic* approach to SHM system design based on WSNs. Specifically, we make the following contributions in this paper. (1) We present the design of a damage localization system that integrates a decentralized computing architecture optimized for the *Damage Localization Assurance Criterion* (DLAC) algorithm [2], [3]. In contrast to centralized approaches that require transporting large amounts of sensor data to a base station, our decentralized architecture pushes the execution of portions of the damage localization algorithm onto each sensor. This in-situ processing results in significant reductions in communication overhead and energy consumption. (2) We also present a proof-of-concept implementation of this design using the TinyOS operating system [4]. In contrast to earlier WSN systems that focus on data collection, our system can detect and localize damages while consuming only a small faction of resources available on the Intel Imote2 [5], an off-the-shelf sensor platform. (3) We provide empirical results and analysis that demonstrate that DLAC can accurately detect and localize damage on a simple beam structure and on a complex truss structure, and that our decentralized approach significantly outperforms a centralized approach in terms of latency, energy efficiency, and system

lifetime. Our work provides an example of the key advantages of a holistic approach to cyber-physical systems.

We begin by discussing related SHM and WSN systems in Section II. Section III presents the design and implementation of our damage localization system. In Section IV, we demonstrate that this system can effectively locate damage to two different physical structures. Section V provides an empirical analysis of the advantages and efficiency of our system on the Imote2 platform. Finally, we conclude in Section VI.

II. Related Work

During the last several years, the structural engineering community has pursued the development of analytical methods to detect and quantify structural damage as well as reliable sensing technologies [6]–[9]. WSNs are gaining the attention of structural engineers as an attractive tool due to their on-board processing and relatively low capital and maintenance costs [10]–[12]. A survey of academic and commercial wireless sensor platforms can be found in [13].

Extensive research in the structural engineering field has focused on developing sophisticated and fault tolerant algorithms for damage detection [13], [14]. These techniques are generally centralized, requiring computations involving global information (usually acceleration data) collected at a single location, e.g., at the base station. With potentially hundreds of nodes and sampling frequencies of hundreds of Hz, these centralized approaches exhibit high energy costs and long delays due to communication overhead.

A schematic paradigm for distributed wireless monitoring system is discussed in [15], [16]. SHM approaches using a distributed computing strategy have been validated on a scale three-dimensional truss model [15], [17] using algorithms described in [18], [19]. These works address the problem primarily from a structural engineering and algorithmic perspective. In contrast, we propose a *holistic* approach to designing and optimizing a decentralized computing architecture based on the characteristics of a practical damage localization algorithm. Moreover, our paper presents an in-depth analysis of the feasibility and advantages of our decentralized computing architecture in terms of latency, energy consumption, and system lifetime.

In the area of sensor networks, Wisden [20], [21] provides services for reliable multi-hop transmission of raw sensor data, using run-length encoding to compress the data before transmission. A UC Berkeley project to monitor the Golden Gate Bridge [22]–[24] is considered to be the largest deployment of wireless sensor networks for SHM purposes. Vibration data is collected and aggregated at a base station under a centralized network architecture, where frequency domain analysis is used to perform modal content extraction. It takes nearly a full day to transmit sufficient data for such computations, creating latencies that would be inadequate for damage detection after extreme events (e.g., an earthquake). BriMon [25] partially addresses the communication bottleneck by sampling data at 400 Hz and averaging this data over 40 Hz windows. The data resolution and network size (a maximum of 12 nodes per span) supported by BriMon may not be fine-grained enough for damage detection and localization on complex structures. All three of these projects focus primarily on data collection and networking challenges, and rely on a central base station to perform actual damage detection. In contrast, our system features a decentralized architecture that exploits processing on each sensor, achieving significant improvements over a centralized approach in terms of latency, energy efficiency, and lifetime. Moreover, we provide empirical results that demonstrate that our system can effectively localize damages on physical structures, while none of the above papers present results on damage detection or localization.

III. Design and Implementation

In this section, we describe our SHM system designed based on a holistic approach. We first present a damage localization algorithm that is particularly suitable for decentralized processing on wireless sensors. We then describe a decentralized architecture specifically optimized for this damage localization algorithm. A salient feature of our architecture is the partitioning of the damage localization algorithm between the wireless sensors and the base station, which significantly reduces the sensors' communication load and energy consumption in exchange for moderate processing costs on each sensor. We also discuss an implementation of our system and the system challenges that we have overcome during this implementation effort.

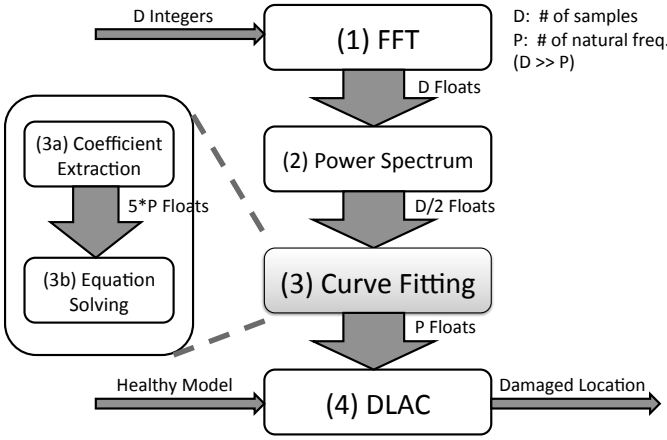

Fig. 1. The online phase of damage localization

A. Damage Localization Algorithm

Our system is based on the *Damage Localization Assurance Criterion* (DLAC) technique [2], [3], which analyzes data collected at each sensor to detect and localize structural damage. The DLAC algorithm is especially well-suited for a decentralized WSN system [26], [27], because it performs damage localization based on post-processed natural frequency data rather than raw vibration data. As discussed below, this natural frequency data is computed from each node's raw vibration data (i.e., accelerometer readings). In Section III-B, we discuss how this computation can be appropriately partitioned between

the base station and sensor nodes, significantly reducing the communication and energy burden in exchange for moderate in-situ processing. Moreover, nodes do not need to correlate individual sensor readings to compute this natural frequency data. Existing systems based on time-domain analysis require precise time synchronization across nodes, incurring additional communication and energy overhead [20], [24].

In the rest of this subsection, we will summarize the damage localization procedure. The damage localization process includes an offline phase and an online phase. In the offline phase, the system identifies the natural frequencies of the healthy structure, using observed vibration (acceleration) data and a series of transformations described below. Because these natural frequencies change in response to structural damage, they are an effective "signature" of the structure's health. (We note that the natural frequencies are uniform throughout the entire structure, and so even localized damage will produce a global change in the frequency content.) Additionally, as required by the DLAC technique, an analytical model of the structure and the estimation of its natural frequencies using purely numerical techniques are performed[1].

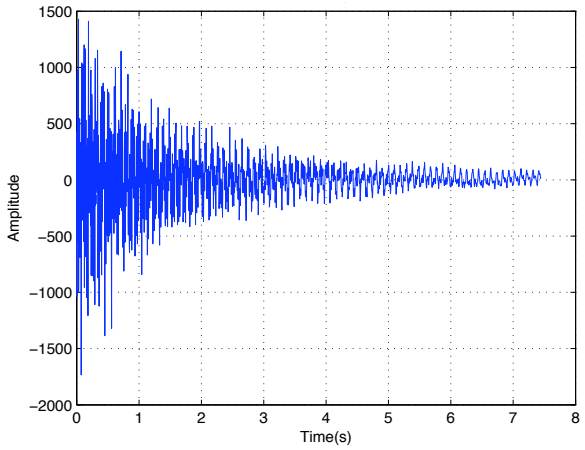

Fig. 2. Raw vibration readings taken after exciting a steel beam with a hammer

In the system's online phase, we periodically sample new vibration data. An example of a raw sensor reading, taken during the experiment described in Section IV-A, is shown in Figure 2. We then repeat the natural frequency identification techniques on this newly-collected data. In the final stage of the algorithm, this new frequency data and the structure's analytical model enable the DLAC algorithm to localize the damage to discrete locations on the structure.

The online phase of our system can be decomposed into four stages, which are summarized in Figure 1. Steps (1)-(3) are used to compute the current natural frequencies of the structure based on collected vibration data, which are then input into the DLAC algorithm in Step (4).

[1]The details of the model's creation, as well as these numerical techniques, are well-established in the structural engineering field and are beyond the scope of this paper.

(1) The raw sensor readings are converted from time domain data to frequency domain data using a **Fast Fourier Transform** (FFT). This produces a series of complex numbers as output, represented as an array of floating point numbers twice the length of the original input (one real and one imaginary part per input). A property of the FFT output data is that its magnitudes are symmetric. To save memory and computation in later stages, we discard the redundant half of this frequency domain data, resuting in a final output the same length as the input.

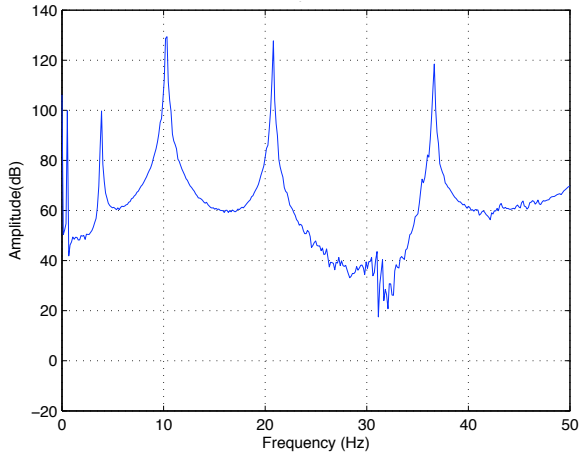

Fig. 3. Power spectrum analysis results of raw vibration data, with the redundant upper half already removed

(2) The FFT's output is fed into a **power spectrum analysis** routine, which calculates the squared magnitude of each complex value in the FFT output data. Figure 3 demonstrates the output of power spectrum analysis over the previous raw sensor data trace.

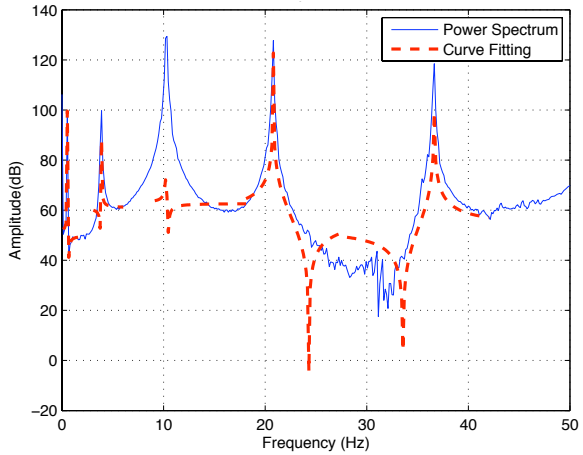

Fig. 4. Polynomial curve fit to the power spectrum analysis data

(3) We can then identify the natural frequencies in this power spectrum data by performing **polynomial curve fitting**. The goal of this process is to identify the frequencie values associated with the peaks in the power spectrum curve for

each mode. Empirical study has shown that the Fractional Polynomial Curve-Fitting (FPCF) technique is reliable for identifying a structure's modal frequencies in an automated manner. FPCF fits the power spectrum data to a polynomial function in the form of Equation 1, with the order of its denominator proportional to the number of frequencies we wish to locate. This function was identified in [28] to extract features from system transfer functions, and represents both a smoothing and an interpolation of the raw power spectrum data.

$$H(s) = \frac{B(s)}{A(s)} = \frac{b_1 s^m + b_2 s^{m-1} + \ldots + b_{m+1}}{a_1 s^n + a_2 s^{n-1} + \ldots + a_{n+1}} \quad (1)$$

Figure 4 illustrates the results of fitting a 2nd-order curve near each seperate peak in the power spectrum data discussed above. We note that, as in Figure 4, the fitted curve does not necessarily match the amplitude (Y-axis) of the power spectrum data at all of the peaks. The goal of this step is to obtain the imaginary parts of the roots of Equation 1's denominator, which correspond to the frequencies of the structure; the amplitude of the fit is therefore irrelevant.

For the purposes of implementation and analysis, we subdivide the identification of natural frequencies into two steps: (3a) **coefficient extraction**, which represents the curve-fitting problem as a series of matrices; and (3b) **equation solving**, which applies the matrix operations necessary to determine the roots of the denominator polynomial.

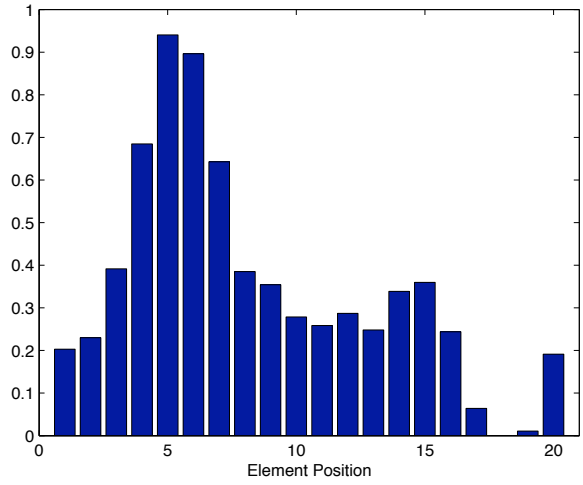

Fig. 5. DLAC results representing the correlation of damage to 20 discrete locations along a steel beam; higher numbers represent a greater likelihood of damage

(4) Finally, once the structure's natural frequencies have been identified, they are used as input into the **DLAC** algorithm, which ultimately detects and localizes damage to the structure. Based on these inputs, DLAC yields a vector of numbers in the range $[0, 1]$, representing the correlation factors to damage at various discrete locations along the structure. A concentration of relatively high values indicates strong correlation and therefore a potential damage location.

The DLAC algorithm is performed as follows. Offline, steps (1) through (4) are executed when the structure is known to be healthy. Using the coefficients of $A(s)$ in Equation 1, we identify the vector $\omega_{healthy}$ that represents the structure's natural frequencies in its healthy state. Using the structure's numerical model, we also estimate the structure's natural frequency vector $\omega'_{healthy}$ using purely numerical techniques.

This numerical model is also used offline to simulate damage at discrete locations along the structure, providing an estimate of what the natural frequencies *would* be if the structure were damaged at each of these locations. We say that the vector ω_j predicts the structure's natural frequencies when damage is simulated at location j. For each ω_j, we calculate a *frequency change* vector $\delta\omega_j$, where

$$\delta\omega_j = \frac{\omega'_{healthy} - \omega_j}{\omega'_{healthy}} \quad (2)$$

We note that $\delta\omega_j$ is normalized with respect to $\omega'_{healthy}$; this normalization gives equal weight to all vectors and reduces any bias induced by higher modes. It is also worth emphasizing that, because $\delta\omega_j$ is calculated relative to the predicted $\omega'_{healthy}$ rather than the observed $\omega_{healthy}$, the final results will be relatively robust to imperfections in the numerical model.

Steps (1) through (4) are then repeated online, giving a new frequency vector ω_{damage}. We likewise compute a frequency change vector $\Delta\omega$ for this data, i.e.,

$$\Delta\omega = \frac{\omega_{healthy} - \omega_{damage}}{\omega_{healthy}} \quad (3)$$

Finally, we compute the correlation between the actual change in frequency, $\Delta\omega$, and each predicted change in frequency, $\delta\omega_j$, as

$$DLAC_j = \frac{(\Delta\omega \bullet \delta\omega_j)^2}{|\Delta\omega|^2 \cdot |\delta\omega_j|^2} \quad (4)$$

In Figure 5, we plot $DLAC$ for a steel beam that has been subdivided into 20 discrete regions; relatively high DLAC values concentrated around $X = 5$ indicate a strong correlation with damage at the fifth region.

A salient feature of DLAC is that it ultimately represents hundreds or thousands of raw sensor readings as a single vector ω_{damage}. As we discuss in Section V, this representation effectively compresses the data by up to 99.8% in a typical SHM setup, significantly reducing the network's communication burden. This is an especially attractive feature for wireless sensor networks, where wireless bandwidth is often limited and sensors typically have a low energy budget. However, we note that DLAC is designed to detect damage at only one location; other techniques are needed to detect multiple damage locations [29], which we plan to explore as future work.

B. Decentralized Architecture

We have developed a decentralized computing architecture specifically optimized for the damage localization procedure

presented in Section III-A. Our structural health monitoring system consists of low-power sensors (also called *motes*) and a base station connected by a wireless network. Motes typically have limited resources (e.g., processing capabilities and memory) and run on batteries. Due to the difficulty of replacing batteries for sensors embedded in a structure, the sensors' energy efficiency is a critical concern for SHM systems. In contrast, the base station (typically a PC) is connected to a wired power source and has significantly more resources than the sensors. Each mote collects raw vibration data from an attached accelerometer and performs parts of the damage localization procedure. The motes transmit their partial results wirelessly to the base station, which completes the damage localization procedure.

With the advance of sensor hardware, commercial sensor platforms such as the Imote2 are capable of moderate amounts of in-network processing. Our decentralized architecture exploits these processing capabilities to reduce the communication and energy costs of damage localization. Because portions of the damage localization procedure described in Section III-A (e.g., the DLAC algorithm) involve complicated optimization routines, it is impractical to perform damage localization entirely on the motes. However, offloading too much computation onto the base station would require transmitting large amounts of data, on the order of thousands of floating-point numbers. An important design goal of our system was therefore to find the proper balance between the time and energy spent on computations on the motes, and the time and energy spent sending partial results to the base station.

To identify the optimal partitioning between the motes and the base station, we analyze here the data flow between stages of the damage localization procedure. We validate our analysis through a comprehensive empirical measurement of different partitioning strategies in Section V. As shown in Figure 1, we parameterize this analysis by the number of samples being collected, D, and the number of frequencies to identify, P ($D \gg P$). The FFT stage consumes D integer sensor readings as input, and produces D floating-point values as output. Power spectrum analysis transforms these D floating-point values into $\frac{D}{2}$ floating-point magnitudes. The coefficient extraction portion of the curve-fitting routine represents the power spectrum data as $5P$ floating-point coefficients; applying the equation solver reduces this to P floating-point values.

As shown by the detailed empirical evaluation in Section V, partitioning between the curve fitting and DLAC stages results in an optimal energy efficiency and latency. The curve fitting routine results in significant reduction in the amount of data that must be transferred to the next stages, from the hundreds or thousands of collected vibration samples to a single vector of size P. For a typical setup of $D = 2048$, $P = 5$, 16-bit accelerometer readings, and single precision (32-bit) `float` types, the stages before curve fitting generate from 4 KB to 16 KB of data; in comparison, curve fitting outputs only 20 B. In practice, the relatively complex equation solving substage of the curve fitting routine may be impractical to implement on some sensor network platforms. The system may alternatively be partitioned between the coefficient extraction and equation solving substages of the curve fitting routine, which outputs $5P$ matrix coefficients (100 B of data under the setup described above). Based our detailed empirical analysis described in Section V, the in-situ processing performed before either partitioning point reduces the communication latency so that the raw data collection stage dominates the algorithm's running time. Similarly, the radio's energy consumption is then dwarfed by the cost of idle sleeping when either partitioning point is selected, and represents 0.98% or less of the system's total energy budget. This partitioning of the damage localization procedure between the motes and the central base station highlights the importance of an integrated design for the computing architecture and the damage localization techniques.

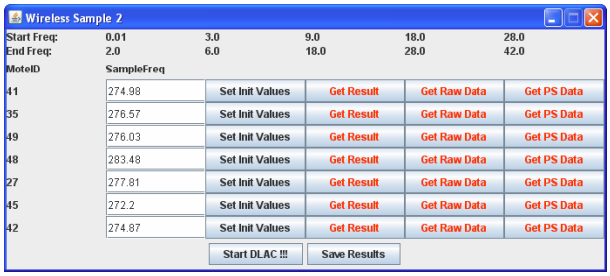

Fig. 6. The damage localization user interface

C. Implementation

Our architecture is implemented as a proof-of-concept SHM system containing two major software packages, which are available as open-source software at [30]. The first package is implemented on top of the TinyOS 1.1 operating system, and is deployed on the Imote2 hardware platform. The Imote2 motes are equipped with 32 MB of RAM, XScale CPUs capable of running at speeds up to 614 MHz, and add-on sensor boards with integrated accelerometers [31].

Our current implementation assumes that sensors are within a single hop from the base station, as the focus of this work is on decentralized processing rather than network protocols. However, our system can easily be extended to support multi-hop networks by incorporating existing multi-hop data collection protocols [24], [32]. We discuss the implications of multi-hop networking on our system's lifetime in Section V-D.

The second software package consists of a Java application and MATLAB scripts running on the base station PC. A GUI (shown in Figure 6) allows users to set the algorithm's parameters, initiate data collection and aggregation on individual motes, and collect the partial curve fitting results computed by the motes. Once the application receives partial results from a mote, it completes the curve fitting procedure using an equation solver written in Java. The results of this equation solver are then processed using a MATLAB script that implements the DLAC algorithm. For debugging purposes, our system can also retrieve the last set of raw sensor readings from individual motes; this feature is not used under normal operations.

To simplify the implementation, the SHM algorithm is currently invoked only when requested by the PC-side GUI. The motes currently keep their radio on to listen for these control messages, which can rapidly deplete their batteries. We emphasize that there is nothing *inherent* in our decentralized approach that prohibits performing autonomous readings at prescheduled intervals and/or managing the radio power, e.g., by using existing power-efficient MAC protocols. We discuss these options in greater detail in Section V-D.

D. Implementation Challenges

Sampling Jitter: One important lesson that we encountered early in our project is the significant impact of jitter in sensor sampling intervals on damage localization. We initially targeted the Imote1 platform for our system but observed poor experimental results. We traced the poor results back to the Imote1's sensor board, which sampled the accelerometer at highly variable intervals. The significant jitter in the sampling interval resulted in poor damage localization results, even though the damage localization procedure itself was implemented properly. We attempted to debug the Imote1's sensor drivers but were hindered by the fact that they are partially closed-source.

After switching to the Imote2 platform, we discovered other, smaller inaccuracies our experimental results. The accelerometer chip on the Imote2's ITS400 sensor board can be programmed to collect samples at discrete frequencies of 280 Hz, 560 Hz, 1120 Hz, or 4480 Hz. Using an oscilloscope, we determined that their sensor chips deviated within $\pm 10\%$ of their programmed frequencies. While the "actual" sensing frequencies varied from board to board, we did not observe variations in frequency over time for individual boards within our controlled lab environment; e.g., a board programmed to sample its accelerometer at 560 Hz might actually operate at 550 Hz, but it would *consistently* operate at 550 Hz. For the purposes of our proof-of-concept implementation, we therefore simply measured the real sampling frequency of each board offline using an oscilloscope and used this calibration data as input to the power spectrum analysis routine. An autonomous or semi-autonomous system could perform this calibration online using the motes' onboard microsecond clock.

Sensing Noise: After performing initial experiments on the truss structure, we discovered that our results were not as high-quality as on the simpler beam structure. We determined that the truss's more complex geometry introduced noise into the sensor readings that degraded the DLAC results. Additionally, a 280 Hz sampling rate was insufficient to identify the higher frequencies in this structure. As a result, we increased the frequency of data collection from 280 Hz to 560 Hz and performed averaging over five consecutive sets of readings.

IV. Evaluation: Damage Localization

In this section, we present an evaluation of our SHM system's *physical* performance, discussing our system's ability to localize damage on two sample structures. The two structures' different physical properties serve as good indicators of DLAC's performance under ideal and complex conditions, respectively.

Mode	1	2	3	4	5
Measured	0.5381	4.0240	11.4705	22.5506	37.4316
Analytical	0.6564	4.1133	11.5180	22.5710	37.3160

TABLE I
MEASURED AND ANALYTICAL NATURAL FREQUENCIES FOR THE HEALTHY BEAM

Mode	1	2	3	4	5
Analytical	0.6555	4.0105	10.6192	20.8768	36.1469
Sensor 1	0.5506	3.9043	10.2473	20.7641	36.6415
Sensor 2	0.5374	3.8902	10.2779	20.8069	36.6396
Sensor 3	0.5402	3.8977	10.2714	20.7964	36.6048
Sensor 4	0.5316	3.8564	10.2744	20.8470	36.6785
Sensor 5	0.5371	3.7678	10.0707	20.4038	36.9797
Sensor 6	0.5427	3.8488	10.3217	20.7546	36.5919
Sensor 7	0.5392	3.9012	10.2533	20.7751	36.6570

TABLE II
ANALYTICAL AND IDENTIFIED NATURAL FREQUENCIES FOR THE DAMAGED BEAM

A. Beam

To validate our damage localization system, we first performed a series of experiments on a steel cantilever beam in the Structural Control and Earthquake Engineering Lab at Washington University in St. Louis. The beam, depicted in Figure 7, is 2.75 m long, 7.6 cm wide, and 0.6 cm thick and fixed to the ground to approximate a cantilever support. Damage along the beam can be simulated at three distances from the beam support by attaching a 1.5 kg steel bar. Because this beam has relatively simple structural properties, it serves as a test of our system under ideal conditions.

We collected data about the beam's healthy state by attaching seven Imote2 wireless sensors at equidistant intervals along the beam. Each mote was equipped with a Crossbow ITS400 sensor board with embedded 3-axis accelerometers; tests on a shake table confirmed that these accelerometers are sufficiently accurate for DLAC purposes within their saturation range of $\pm 2.0g$. After exciting the beam with a hammer, we collected vibration data from each mote. Using this data, we determined the beam's healthy natural frequencies offline, as shown in Table I.

A corresponding 2D Bernoulli beam model was generated in MATLAB, which subdivided the beam into 20 elements with 42 global degrees of freedom (Figure 8). As shown in Table I, the first natural frequency predicted by the model is within 22% of the experimental value, while the other predicted frequencies fall within 2% of the experimental data. These discrepancies can be explained by simplifying assumptions in the model; e.g., the Imote2 nodes were not included in the model. We remind the reader that the DLAC algorithm uses both measured data and analytical data as inputs, thus accounting for such discrepancies.

We then tested our system's ability to detect and localize damage along the beam structure. Using the procedure described in Section III, we collected and analyzed vibration data at 280 Hz, both in its healthy condition and with the steel bar

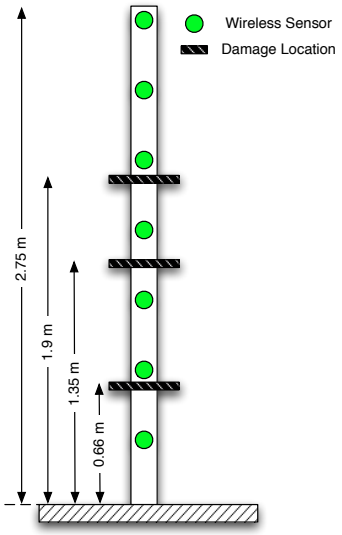

Fig. 7. Diagram of cantilever beam test structure

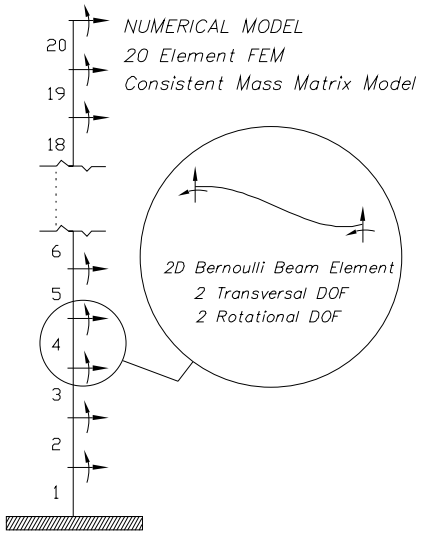

Fig. 8. Cantilever beam finite element model

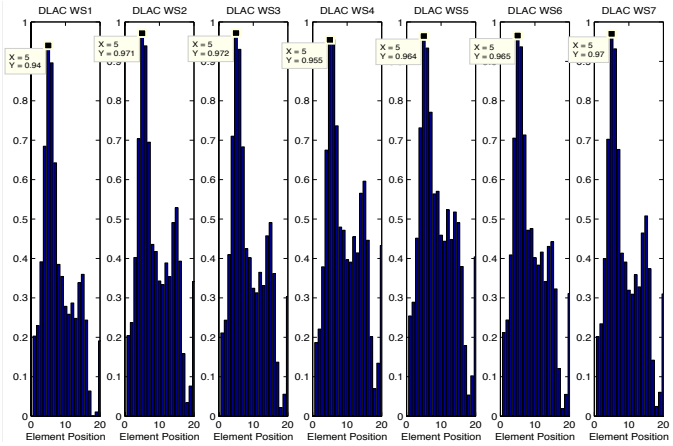

Fig. 9. DLAC results for the beam damaged at element 5

attached at each of the three damage locations shown in Figure 7. We added an arbitrary amount of mass at each position in our analytical model to develop the matrix of damage cases for computation of the correlation factors. The amount of mass that we added to the model intentionally did not match the steel bar's actual mass. We included this discrepancy to reflect the fact that the amount of damage to a structure is not known ahead-of-time, and to illustrate that DLAC will still adequately localize damage as long as a reasonable guess is used.

For the sake of brevity, we present here only the results for the first scenario, which simulates damage at the beam's fifth element. As shown in Table II, the natural frequencies measured by each of the 7 sensor nodes closely match those predicted by the "damaged" analytical model. Each node therefore correctly predicts structural damage at the beam's fifth element with a correlation of 94% or higher (Figure 9). We observed similar results during the other two damage scenarios, with the nodes consistently localizing the damage at the correct element with correlations of 90% or higher.

Fig. 10. 3D truss test structure

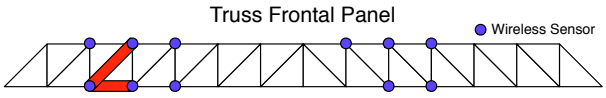

Fig. 11. Truss experimental setup; highlighted elements were replaced to simulate damage

B. Truss

To evaluate our system under more complex structural configurations, we then performed tests on a 5.6 m steel truss structure [33] at the Smart Structure Technology Laboratory (SSTL) at the University of Illinois at Urbana-Champaign (see Figure 10). 11 Imote2 sensors were deployed on the frontal panel of the truss, as shown in Figure 11; USB cabling was deployed to power the motes, but all communication occurred over their wireless radios. The truss consists of

Mode	1	2	3	4	5
Measured	20.65	41.49	64.59	69.41	95.51
Analytical	19.88	38.31	66.26	67.17	92.25

TABLE III
MEASURED AND ANALYTICAL NATURAL FREQUENCIES FOR THE HEALTHY TRUSS

Mode	1	2	3	4	5
Analytical	19.19	38.35	63.58	66.30	90.96
Sensor 1	20.27	41.37	63.04	67.79	94.89
Sensor 2	20.28	41.40	63.17	67.89	95.08
Sensor 3	20.20	41.29	63.01	67.67	94.82
Sensor 4	20.17	41.23	63.05	67.68	94.73
Sensor 5	20.31	41.30	63.10	67.73	94.89
Sensor 6	20.23	41.29	63.02	67.68	94.81

TABLE IV
ANALYTICAL AND IDENTIFIED NATURAL FREQUENCIES FOR THE DAMAGED TRUSS

fourteen bays 0.4 m-long bays and sits on four rigid supports. Different structural configurations and damage scenarios can be emulated by removing or replacing the truss's members and its supports.

As with the beam, we used collected truth data and a MATLAB model to compute the natural frequencies in the truss's healthy state. We collected the truth data by vertically exciting the truss structure using a magnetic shaker. (To ensure a consistent mass distribution with later experiments, the Imote2 motes were left installed but were not activated.) A force transducer was used to measure the input force, and six wired sensors were used to measure the vibrations at different points on the truss's frontal panel. A corresponding numerical finite element model with 160 beam elements and 336 global degrees of freedom (Figure 12) was generated in MATLAB. As shown in Table III, the natural frequencies predicted by this model are within 2–7% of the experimental data. Again, these discrepancies can be explained by simplifying assumptions in the model and are accommodated by the DLAC algorithm.

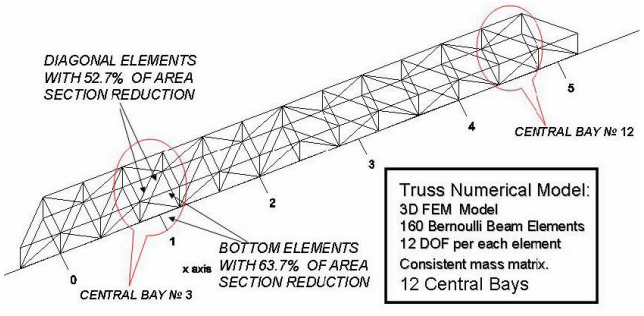

Fig. 12. Truss finite element model

To simulate damage along the truss structure, we replaced the beam elements of the third bay (highlighted in Figure 11) with smaller elements. Specifically, two diagonal elements were reduced in area by 52.7%, and two bottom elements were reduced in area by 63.7%. We simulated damage to the truss's numerical model by reducing the model's corresponding beam elements.

We then excited the "damaged" truss structure and used the Imote2 nodes to collect vibration data. Because the truss has more complex behavior than the beam, we increased the sampling frequency to 560 Hz. To reduce noise, we also averaged the power spectrum results over five consecutive readings. 6 of the 11 sensors reported enough vibration data[2] to compute natural frequencies with a DLAC correlation of 85%. The natural frequency data and DLAC results are shown in Table IV and Figure 13, respectively. The DLAC results strongly predict damage in the third bay, which is where the elements were replaced.

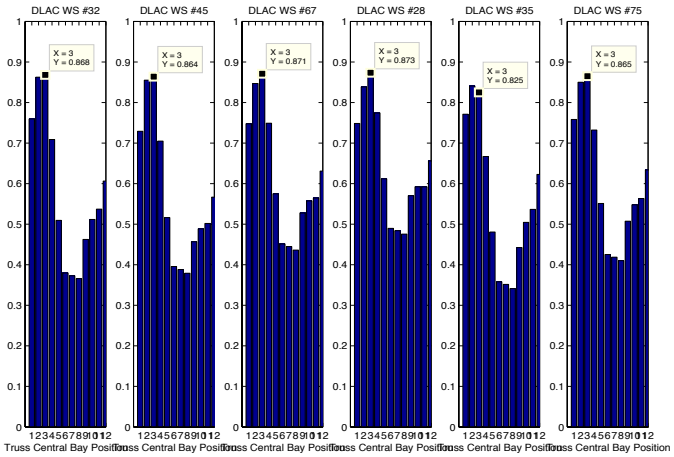

Fig. 13. DLAC results for the damaged truss

V. EVALUATION: CYBER SYSTEM PERFORMANCE

We now evaluate the *cyber* aspects of our cyber-physical SHM system. First, we will validate the optimal partioning of the decentralized algorithm proposed in Section III-B, by showing that it outperforms other potentional partitioning points in terms of latency and energy consumption. Second, we will demonstrate that our optimally-partitioned system significantly outperforms a centralized approach in terms of system lifetime. Based on these findings, we project that our system would achieve a lifetime of approximately 193 days between battery replacements with appropriate power management techniques.

Throughout this section, we will consider five different configurations of our system. Four of these five configurations represent different partitionings of the decentralized algorithm discussed in Section III-B: they respectively perform up to the FFT, power spectrum analysis, coefficient extraction, and equation solving stages on the mote before transmitting their partial results to the base station. The fifth configuration performs no computations and transmits its raw sensor data back to the base station, representing the behavior of a fully centralized application.

A. Memory

Figure 14 presents the ROM consumption of five different configurations of our SHM system. The onboard FFT routine

[2]The Imote2 vibration sensor will occasionally fail to collect a round of samples, due to a driver bug that could not be isolated by the time the experiments were run.

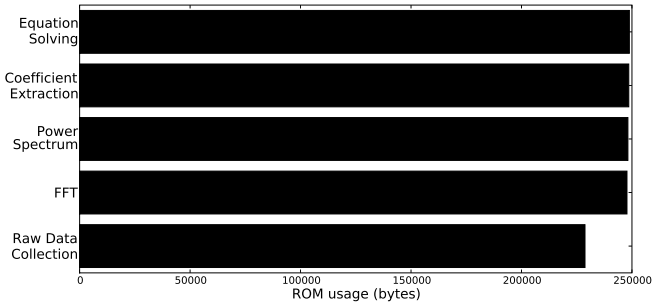

Fig. 14. The ROM footprint of different SHM system configurations

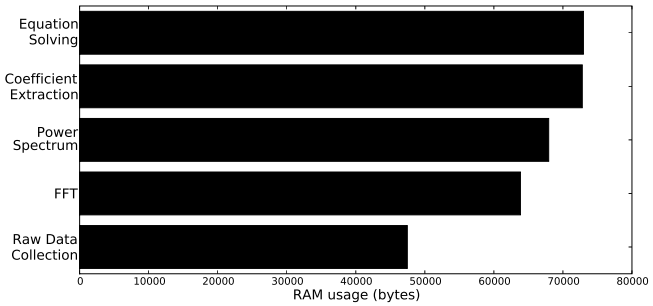

Fig. 15. The RAM footprint of different SHM system configurations

has the largest impact on footprint, increasing the size of the application from 228748 bytes to 247748 bytes (8.3%), while the other routines add between 264 bytes (1.1%) and 424 bytes (1.7%) each. We see a larger difference in RAM consumption as we increase the amount of onboard computation, as shown in Figure 15. The FFT routine again increases the footprint the most, from 47460 bytes of RAM to 63844 bytes (34.5%). The remaining routines require an additional 166 bytes (0.2%) to 4864 bytes (7.1%).

In absolute terms, this footprint fits well within the hardware capabilities of the current-generation sensor hardware. Indeed, on platforms such as the Imote2 (which is equipped with 32 MB each of flash ROM and SDRAM) this application would significantly *under*utilize the hardware capabilities. As shown above, the incremental cost of adding each additional onboard computation is also small in relative terms. Nevertheless, the memory consumption of our system could be further reduced by two straightforward optimizations, which could potentially expand the number of platforms which our system could be deployed on.

First, because our application was designed for the relatively resource-rich Imote2 platform, we have not written our codebase with RAM conservation in mind. Specifically, our application retains copies in RAM of the raw sensor data and the output of intermediate computations. This decision simplifies the implementation and allows us to retrieve these intermediate values for debugging purposes. On more RAM-constrained devices, our application could be altered to keep only a single memory buffer and perform all computations in-place on this single buffer.

Second, the beta Imote2 toolchain for TinyOS 1.1 tends to greatly inflate the footprint of compiled applications compared to other platforms. The Wasabi GCC compiler used by this toolchain will crash unless the toolchain is invoked in debug mode, which disables nesC's aggressive inlining optimizations and inserts debugging symbols into the binary. Also, because binary size is not generally a concern on the Imote2, the toolchain automatically includes complex subsystems (such as a USB debugging console) which contribute to the size of the binary. For comparison, a simple test application included in TinyOS (`CntToRfm`) consumes 195052 bytes of ROM and 18532 bytes of RAM on the Imote2 platform compared to 11234 bytes of ROM and 371 bytes of RAM for the TelosB platform. We anticipate that deploying our application with a different toolchain (whether a different platform or the more modern, stripped-down Imote2 toolchain used by TinyOS 2.1) would therefore achieve a significant footprint reduction.

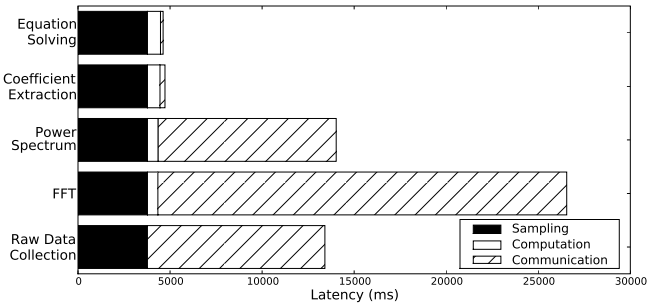

Fig. 16. The latency of sensor data collection and processing

B. Latency

To evaluate the latency of a single round of damage detection, we timed the execution of its constituent steps: collecting the raw sensor data from the accelerometer, performing onboard computations on the data, and transmitting the computed results back to the base station. Again, because the computation and communication latency of our SHM system depends greatly on how much computation is performed onboard, we present this data for the five different system configurations. Where possible, we measured these latencies using the Imote2's onboard microsecond timer and took the mean of 50 rounds. Because the Imote2's onboard radio interferes with the hardware microsecond timer, the data transmission latencies (with the exception of the FFT data latency[3]) were collected over 10 rounds using an oscilloscope. We focus here on the latencies incurred by on-board processing and communication, excluding processing at the base station. We note that this decision benefits the fully centralized approach, which will pay a comparatively higher processing cost at the base station.

Figure 16 presents the average latency for each of these five configurations. All five schemes incur a mean cost of 3772 ms

[3]Our oscilloscope did not have a large enough data buffer to reliably measure the time spent transmitting the FFT data. We instead measured this latency by instrumenting the PC base station software, which we expect to provide results within one packet RTT of the actual time spent transmitting.

(σ = 0.80 ms) to collect raw sensor data. This closely matches the $\frac{2048}{560\,\text{Hz}} \approx 3.7$ s needed to collect 2048 samples, with some additional overhead to copy the sensor data into a local buffer. The cost of all the onboard computations is relatively small: the FFT, power spectrum analysis, coefficient extraction, and equation solving routines consume 566.8 ms (σ = 2.78 ms), 17.1 ms (σ = 2.78 ms), 97.2 ms (σ = 0.01 ms), and 27.1 ms (σ = 0.26 ms) respectively.

These latter two computations reduce the data to be transmitted by 98.8% and 99.8% respectively, from 2048 data points to 25 and 5. Therefore, these two configurations take only 271 ms (σ = 11 ms) and 142 ms (σ = 16 ms) respectively to transmit their results to the base station, compared to the 9638 ms (σ = 28 ms) to transmit all raw data in the fully-centralized case. By performing computation and an appropriate amount of processing on the nodes, we incur very little system overhead on our current-generation sensor hardware. 79.8% to 81.6% of the system's time is spent collecting data; only 20.1% or less of the latency represents reducible overhead. In comparison, the centralized approach spends 71.9% of its time transmitting data to the base station. As a result, our decentralized system can achieve latencies up to 65.5% lower than those of a centralized algorithm. It is also worth noting that delegating the equation solving substage to the base station incurs only a 2.2% performance penalty compared to doing the entire curve-fitting routine onboard, because both approaches are dominated by the time spent collecting raw sensor data. Therefore, transmitting the partial curve-fitting results is an acceptable alternative on systems where the equation solving routine cannot realistically be implemented.

Notably, performing the power spectrum analysis onboard does not reduce latency at all, and performing FFT onboard is actually counterproductive: it takes 22206 ms (σ = 133 ms) to transmit the FFT results and 9668 ms (σ = 28 ms) to transmit the power spectrum data to the base station. This phenomenon validates the data flow analysis in Section III-A (note that the single-precision floating-point values in the FFT and power spectrum data are twice the width of the 16-bit sensor readings). These findings also highlight the importance of a systematic evaluation for identifying the optimal configuration of cyber-physical systems through data-flow analysis and empirical benchmarks.

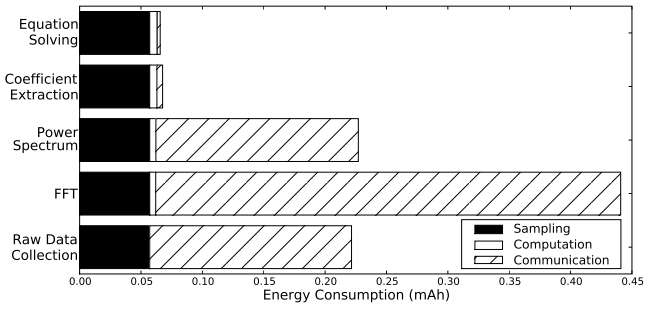

Fig. 17. The energy consumption of sensor data collection and aggregation

C. Energy Consumption

The current version of our SHM system performs only limited power management, since the TinyOS 1.1 drivers for the Imote2 do not put all of the hardware to sleep when deactivated. The Imote2 driver subsystem has been rewritten for TinyOS 2.1, which was released shortly before this writing and which we expect will fix this shortcoming. Nevertheless, we can estimate the energy consumption of a fully power-managing SHM system by combining the latency statistics given above with current consumption data for the radio, sensor, and CPU taken from the corresponding datasheets [5], [34], [35].

Figure 17 shows the energy cost of a single round of SHM data collection. Performing the entire curve-fitting routine onboard compared to a fully centralized approach significant reduces the energy consumption, from 0.222 mAh to 0.066 mAh. This reduction is mainly due to the expense of sending raw sensor readings to the base station. A configuration which performs the curve-fitting routing onboard consumes 0.006 mAh (31 mA [5] for 708 ms) to perform its computations. However, these computations save the node an average of 0.162 mAh during transmission, since it reduces the time that the radio is active and transmitting by 9.5 s. Again, offloading the equation solving portion of this routine to the base station has a minimal effect on energy consumption. The node would save 0.0002 mAh on computation costs but would expend an additional 0.0022 mAh on communication, representing an increase of 3% compared to performing the equation solving onboard. The energy consumption of either of these two decentralized approaches is dominated by the cost of collecting raw sensor data (84.5% and 87% of the total energy consumption), whereas the fully centralized approach spends 74.2% of its energy transmitting the sensor readings back to the base station.

We again find that performing any fewer stages of computation onboard is counter-productive. Performing the FFT and power spectrum analysis locally incurs a computational overhead of 0.005 mAh but does not affect the amount of data being sent back to the base station. As a result, this approach incurs a 2.5% energy penalty compared to the fully centralized approach. Computing only the FFT data onboard performs even worse, since its output is double the size of its input. This approach therefore increases the energy consumption by 99.0% over the fully centralized case.

The memory, latency, and energy consumption benchmarks demonstrate that the optimal partitioning point indeed occurs after the curve-fitting routine, as indicated by the data-flow analysis in Section III-B. These results also validate that, on systems where the full curve-fitting routine cannot realistically be implemented, even implementing a portion of this routine provides substantially better performance than simply sending the raw sensor data to the base station for processing. Again, the performance of the FFT and power spectrum routines highlight the importance data-flow analysis in decentralized cyber-physical applications: in terms of RAM, ROM, latency, and

energy consumption, both partially-decentralized approaches perform *worse* than a fully centralized approach.

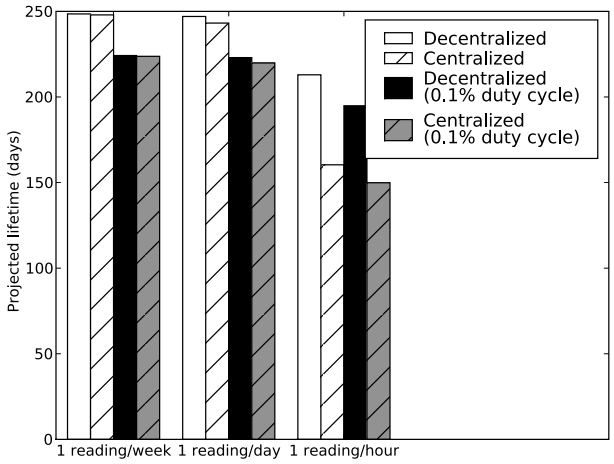

Fig. 18. System lifetime under different usage patterns

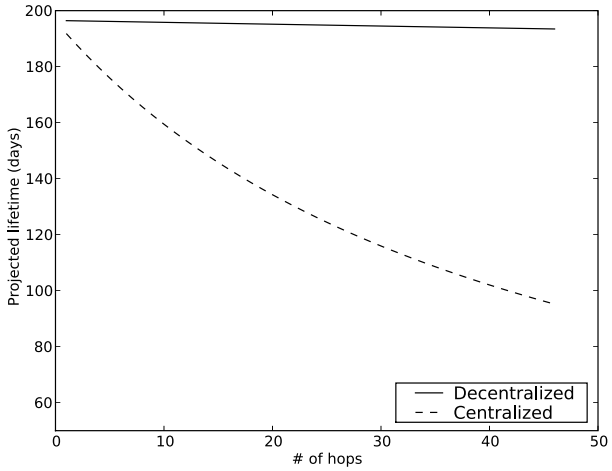

Fig. 19. System lifetime with hourly readings and 0.1% radio duty cycle, under various network configurations

D. Projected Lifetime

We can estimate the system's expected lifetime by noting that the Imote2 consumes 382 μA in its deep sleep state [5], plus 15 μA for the accelerometer [35]. Figure 18 presents the estimated system lifetime when the Imote2 is deployed with a standard 3x AAA battery pack providing 2400 mAh of charge. In the interest of reducing clutter, we only present only the fully-centralized case (i.e., where no processing is performed onboard) and the most decentralized case (where all computations prior to the final DLAC stage are performed onboard). As noted above, performing only the FFT or power spectrum analysis onboard would in fact reduce the node's lifetime, and running only part of the curve-fitting onboard has similar performance to the fully-decentralized case.

If we assume that the system remains asleep between periodic readings, then the decentralized approach achieves a projected lifetime of 213 days, even at a relatively aggressive hourly schedule. In contrast, the centralized approach achieves a lifetime of 160 days at an hourly schedule, though it stays within 0.2% of the decentralized approach's lifetime at lower frequencies. The sharp drop in the centralized system's lifetime occurs because sleeping dominates the system's energy cost at lower frequencies, while the high communications costs dwarf the sleeping cost at an hourly frequency. As a result, in-situ processing enables more frequent monitoring than is realistically possible for a centralized scheme.

In practice, a SHM system may not be able to behave autonomously: its deployers may want some kind of manual control (e.g., to perform on-demand readings after a natural disaster). This can be achieved by having the nodes listen for radio transmissions between readings. Keeping the CPU and radio active at 100% duty cycles would reduce the node lifetime to only 55 hours. However, power-saving MAC layers like SCP [36] can achieve duty cycles as low as 0.1% with reasonable responsiveness tradeoffs. As shown in Figure 18, this would have a fairly low impact on system lifetime (an 8.5%–9.8% reduction in the decentralized case).

The difference in communication costs between a centralized approach and our decentralized approach are amplified under a multi-hop network configuration. This kind of network configuration is necessary for monitoring many real-world structures, since the structure's length will exceed the motes' communication range. For example, [24] required a 46-hop network to span the Golden Gate Bridge, and [25] estimates that 3–4 hops will be needed to span small bridges. The nodes closest to the sink suffer the most from communication overhead, since they must receive and relay packets from all of the nodes further away from the sink. If we assume that nodes are configured in an n-hop line, as in [24], then the node closest to the sink will have to receive $n-1$ sets of data and transmit n sets each time damage detection is performed.

As shown in Figure 19, under the centralized approach this node's lifetime will drop dramatically as the number of hops increases. The mote must keep its radio active for an extra 19.3 seconds for each additional hop, transmitting during half of this time and receiving during the other half. This quickly depletes the mote's battery power, decreasing the network's lifetime from 191 days in a single-hop configuration to 179 days under a 4-hop network, and to 95 days under a 46-hop network. In contrast, the decentralized approach transmits a much smaller amount of data, so that the cost of idle sleeping still dwarfs the communication cost under any realistic hop count. A 4-hop network will reduce the decentralized system's lifetime by 5 hours, and a 46-hop network will reduce the lifetime from 196 days to 193 days. Our decentralized approach therefore represents a 9.2% increase in lifetime under a 4-hop network compared to a centralized scheme, and a 103% increase with a larger 46-hop network.

As observed in [24], reliably transporting large amounts of data over lossy links is challenging. The lifetimes of both approaches will be reduced compared to those projected here, due to packet retransmissions. However, we note that packet

retransmissions will have a significantly higher impact on a centralized system's lifetime, since its communication costs represent a much higher proportion of the total energy budget.

VI. Conclusions

We propose a holistic approach to SHM that features a decentralized computing architecture specifically optimized for the DLAC damage localization algorithm. We have implemented our prototype SHM system on an off-the-shelf sensor platform while using less than 1% of its memory capacity. Our experiments show that, compared to earlier centralized solutions, our system can reduce the latency and energy consumption of each damage localization round by 65.5% and 70.4% respectively, increasing the system's projected lifetime by up to 103% under an hourly schedule. We also demonstrate that our system is able to effectively localize damage to discrete locations on the structure on two physical structures. Finally, we identify the importance of selecting an optimal partitioning point between the onboard processing and the processing done at the base station, through data-flow analysis and systematic empirical benchmarks. These results highlight the advantages of closely integrating the design of computing systems and physical engineering techniques for cyber-physical systems.

Acknowledgment

This work is supported by NSF NeTS-NOSS Grant CNS-0627126. We would like to thank Prof. B.F. Spencer and Shin Ae Jang for the use of the truss for our experiments and all of the valuable assistance provided.

References

[1] S. N. Pakzad, G. L. Fenves, S. Kim, and D. E. Culler, "Design and implementation of scalable wireless sensor network for structural monitoring," *ASCE Journal of Infrastructure Engineering*, vol. 14, no. 1, pp. 89–101, March 2008.

[2] A. Messina, I. A. Jones, and E. J. Williams, "Damage detection and localization using natural frequency changes," in *Conference on Identification in Engineering Systems, U.K.*, 1996.

[3] A. Messina, E. J. Williams, and T. Contursi, "Structural damage detection by a sensitivity and statistical-based method," *Journal of Sound and Vibration*, vol. 215, no. 5, pp. 791–808, 1998.

[4] http://www.tinyos.net.

[5] *Imote2 Hardware Reference Manual*, Crossbow Technology, Inc., 2007.

[6] S. Liu and M. Tomizuka, "Strategic research for sensors and smart structures technology," *Proc. of the International Conference on Structural Health Monitoring and Intelligent Infrastructure, Tokyo, Japan*, vol. 1, pp. 113–117, 2003.

[7] S. Liu and M. Tomizuka, "Vision and strategy for sensors and smart structures technology research," *Proc. of the 4th International Workshop on Structural Health Monitoring, Stanford, CA*, pp. 15–17, 42–52, 2003.

[8] B. Spencer, M. Ruiz-Sandoval, and N. Kurata, "Smart sensing technology: Opportunities and challenges," *Structural Control and Health Monitoring*, vol. 11(4), pp. 349–368, 2004.

[9] C. Farrar, D. Allen, G. Park, S. Ball, and M. Masquelier, "Coupling sensing hardware with data interrogation software for structural health monitoring," *Shock and Vibration*, vol. 13(4), pp. 519–530, 2006.

[10] E. Straser and A. Kiremidjian, "A modular visual approach to damage monitoring for civil structures," *Proc. of SPIE: Smart Structures and Materials*, pp. 112–122, 1996.

[11] A. Kiremidjian, E. Straser, T. Meng, K. Law, and H. Sohn, "Structural damage monitoring for civil structures," *Proc. of Int. Workshop on Structural Health Monitoring, Stanford, CA.*, pp. 371–382, 1997.

[12] E. Straser and A. Kiremidjian, "A modular, wireless damage monitoring system for structures," The John A. Blume Earthquake Engineering Center, Tech. Rep., 1998.

[13] J. Lynch, "Overview of wireless sensors for real time health monitoring of civil structures," *Proc. of the 4th International Workshop on Structural Control*, pp. 189–194, 2004.

[14] H. Sohn, C. Farrar, F. Hemez, D. Shunk, D. Stinemates, and B. Nadler, "A review of structural health monitoring literature: 1996–2001," Los Alamos National Laboratory, Tech. Rep. LA-13976-MS, 2004.

[15] B. Spencer and T. Nagayama, "Smart sensor technology: a new paradigm for structural health monitoring," *Proc. of Asia-Pacific Workshop on Structural health Monitoring, Yokohama, Japan.*, 2006.

[16] J. Lynch and K. Loh, "A summary review of wireless sensors and sensor networks for structural health monitoring," *Shock and Vibration Digest*, vol. 38(2), pp. 91–128, 2006.

[17] Y. Gao, "Structural health monitoring strategies for smart sensor networks," Ph.D. dissertation, University of Illinois at Urbana-Champaign, 2005.

[18] J. Juang and R. Pappa, "An eigensystem realization algorithm for modal parameter identification and model reduction," *J. of Guidance Control and Dyn.*, vol. 8, pp. 620–627, 1985.

[19] D. Bernal, "Load vectors for damage localization," *J. of Engineering Mechanics*, vol. 128(1), pp. 7–14, 2002.

[20] N. Xu, S. Rangwala, K. Chintalapudi, D. Ganesan, A. Broad, R. Govindan, and D. Estrin, "A wireless sensor network for structural monitoring," in *SenSys*, 2004.

[21] J. Paek, K. Chintalapudi, R. Govindan, J. Caffrey, and S. Masri, "A wireless sensor network for structural health monitoring: performance and experience," in *EmNets*, 2005.

[22] S. Kim, "Wireless sensor networks for structural health monitoring," Master's thesis, University of California at Berkeley, 2005.

[23] S. Kim, "Wireless sensor networks for high fidelity sampling," Ph.D. dissertation, University of California at Berkeley, 2007.

[24] S. Kim, S. Pakzad, D. Culler, J. Demmel, G. Fenves, S. Glaser, and M. Turon, "Health monitoring of civil infrastructures using wireless sensor networks," in *IPSN*, 2007.

[25] K. Chebrolu, B. Raman, N. Mishra, P. K. Valiveti, and R. Kumar, "BriMon: a sensor network system for railway bridge monitoring," in *MobiSys*, 2008.

[26] E. Clayton, "Frequency correlation-based structural health monitoring with smart wireless sensors," Master's thesis, Washington University in St. Louis, 2006.

[27] N. Castaneda, F. Sun, S. Dyke, C. Lu, A. Hope, and T. Nagayama, "Implementation of a correlation-based decentralized damage detection method using wireless sensors," in *ASEM Conference*, 2008.

[28] E. C. Levy, "Complex-curve fitting," *IRE Transactions on Automatic Control*, vol. 4, pp. 37–44, 1959.

[29] B. H. Koh and S. J. Dyke, "Structural damage detection in cable-stayed bridges using correlation and sensitivity of modal data," *Computers and Structures*, vol. 85, pp. 117–130, 2007.

[30] http://www.cse.wustl.edu/wsn.

[31] *ITS400 Imote2 Basic Sensor Board*, Crossbow Technology, Inc.

[32] R. Fonseca, O. Gnawali, K. Jamieson, S. Kim, P. Levis, and A. Woo. The collection tree protocol (CTP).

[33] E. Clayton, "Development of an experimental model for the study of infrastructure preservation," *Proc. of the National Conference on Undergraduate Research, Whitewater, Wisconsin*, 2002.

[34] *2.4 GHz IEEE 802.15.4 / ZigBee-ready RF Transceiver*, Texas Instruments.

[35] *LIS3L02DQ MEMS Inertial Sensor*, STMicroelectronics, 2005.

[36] W. Ye, F. Silva, and J. Heidemann, "Ultra-low duty cycle MAC with scheduled channel polling," in *SenSys*, 2006.